自编码器在协同过滤中的应用

QinHsiu[1)]

摘 要 本文对自编码器在协同过滤中的应用进行了概述。论文主要分为三个部分，第一部分分别对基于机器学习的协同过滤和基于自编码器的协同过滤进行了介绍；第二部分通过实验对几个比较经典的模型进行了比较和分析；第三部分对自编码器在协同过滤未来的工作进行了分析与展望并对本文进行了总结。

关键词 推荐系统；协同过滤；自编码器
中图法分类号 TP391 **DOI 号** 10.11897/SP.J.1016.01.2023.00001

Application of Auto-encoder in Collaborative Filtering

QinHsiu[1)]

Abstract This paper provides an overview of the application of self-encoder in collaborative filtering. The paper is divided into three main parts. The first part introduces collaborative filtering based on machine learning and collaborative filtering based on auto-encoder respectively; the second part compares and analyzes several more classical models through experiments; the third part analyzes and outlooks the future work of self-encoder in collaborative filtering and concludes the paper.

Key words Recommendation System; Collaborative Filtering; Auto-encoder

1 引言

随着互联网的快速发展，用户面临着信息过载的问题。大量的信息使得用户难以做出有效的抉择。推荐系统的出现可以有效地缓解这一问题。推荐系统就是一个过滤系统，为用户提供个性化的信息。其不仅改善了用户体验，而且还能因此增加商业利益。协同过滤是推荐系统模型中的一种，其目的是利用用户对于物品的偏好信息（如评价信息）来提供个性化推荐。本文会分别对基于传统机器学习的协同过滤模型和基于神经网络的协同过滤模型进行介绍，侧重主要介绍后者，因为后者在协同过滤中所取得的效果更佳。基于传统机器学习的协同过滤的工作包括矩阵分解[1,2] 和邻域模型[3]，基于神经网络的协同过滤的工作包括基于神经元的模型[4] 和基于自编码器的模型[5–8]。

本文余下部分主要包括三部分，第一部分是对目前协同过滤中基于传统机器学习和基于神经网络的模型的简单介绍；第二部分会通过实验对几个经典模型进行比较和分析；第三部分会对未来工作中自编码器在协同过滤中的应用进行展望并对本文进行总结。

收稿日期：2023-01-06；修改日期：2023-01-06 QinHsiu，None E-mail: **********.

2 研究现状

协同过滤使用用户的偏好数据来进行个性化推荐，偏好数据包括用户对商品的评分或者一些行为（点击、购买等）。在许多面向用户的电子商务和社交媒体的应用中，偏好数据是普遍存在的。按照对偏好数据的使用的方式的不同，可以将协同过滤模型分为：基于传统机器学习和基于神经网络两类。而基于神经网络的模型可以进一步细分为基于自编码器和基于图神经网络两类，本文主要关注基于自编码器的协同过滤部分。

2.1 基于机器学习的协同过滤

表征学习[9] 的目标是为了捕捉和编码观测数据的潜在模式。在偏好数据的情况下，可以将学习到的表征用于推荐。很明显偏好数据是动态的，因为用户每个时刻的偏好是改变的，所以我们需要从这些动态的偏好数据中找到用户与商品之间的关系。为了解决这一问题，潜在因子和矩阵分解模型[1,10–13] 在协同过滤中取得了成功。他们取得成功的主要原因是因为他们简单、有效、高效和易可扩展性。然而这一类的模型受限于其本身只能捕捉数据或潜在空间中的线性模式。

2.2 基于自编码器的协同过滤

随着深度学习的崛起，基于自编码器[5–8]的模型被提了出来用于弥补基于传统机器学习方法在协同过滤中的不足。接下来会对其中比较经典的模型进行一一介绍。

AutoRec[5]是第一个将自编码器用于协同过滤的模型，其模型如图 1所示。对于输入的评分矩阵

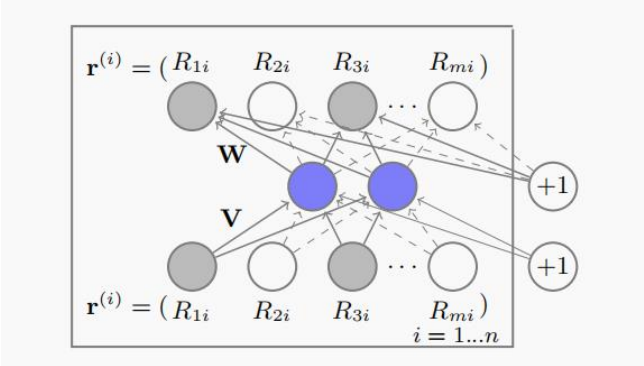

图 1 输入为一个大小为 $|U|\times|I|$ 大小的评分矩阵，经过两个带偏置的全连接层所得的输出为最终的评分矩阵。其中 $|U|$ 是指用户的数目，$|I|$ 是指所有商品的数目，$r^{(i)}$ 表示第 i 个用户对于所有商品的评分的集合，R_{mi} 表示用户 i 对于第 m 个商品的评分。$\mathbf{V}$ 和 $\mathbf{M}$ 分别表示两个全连接层的权重。

r，经过两个全连接层之后，其输出可以使用以下公式表示：

$$h(r;\theta) = f(\mathbf{W}\cdot g(\mathbf{V}r+\boldsymbol{\mu})+\mathbf{b}). \tag{1}$$

其中 $f(\cdot)$ 和 $g(\cdot)$ 是激活函数，$\theta=\{\mathbf{W},\mathbf{V},\mu,b\}$，$\mathbf{W}\in\mathbb{R}^{d\times k}$，$\mathbf{V}\in\mathbb{R}^{k\times d}$，同时偏执 $\boldsymbol{\mu}\in\mathbb{R}^k$，$\mathbf{b}\in\mathbb{R}^d$。
其目标函数如下所示：

$$\min_{\theta}\sum^{n}||r^{(i)}-h(r^{(i)};\theta)||_o^2+\frac{\lambda}{2}\cdot(||\mathbf{W}||_F^2+||\mathbf{V}||_F^2). \tag{2}$$

其优化的目标就是最小化预测值与真实值的均方误差，其中 λ 为超参数，其目的是控制正则项对于模型的影响，该正则是为了防止模型参数过大，同时为了防止模型出现过拟合。

CDAE[6]在 AutoRec[5]的基础上提出为每一个用户分配一个独立的神经元，用于防止模型过拟合，其模型如图 2所示。对于输入的评分矩阵经过第一个全连接层之后所得输出为：

$$z_u = h(\mathbf{W}^T\tilde{y}_u+\mathbf{V}_u+\mathbf{b}), \tag{3}$$

其中 $h(\cdot)$ 为激活函数，$\mathbf{W}$ 为第一个全连接层的权重，$\mathbf{V}_u$ 表示用户 u 的特别表示，$\mathbf{b}$ 为第一个全连接层的偏置。

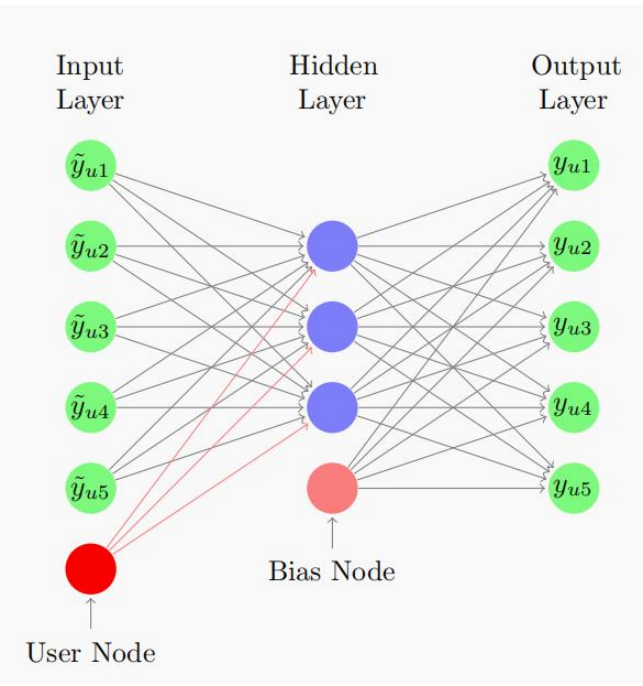

图 2 输入和 AutoRec[5]一样，一个大小为 $|U|\times|I|$ 的评分矩阵，但在经过第一个全连接层之前为每一个用户单独分配了一个神经元（红色表示），经过两个带有偏置的全连接层所得的输出为最终的评分矩阵。其中 $|U|$ 是指用户的数目，$|I|$ 是指商品的数目，$\tilde{y}_{ui}$ 表示用户 u 对于商品 i 的评分。

将经过第一个全连接层的输出 z_u 作为输入，在经过第二个全连接层之后最终的输出为：

$$\hat{y}_{ui} = f(\mathbf{W}_i'^T z_u + b_i'). \tag{4}$$

其中 $f(\cdot)$ 表示激活函数，$\mathbf{W}'$ 表示第二个全连接层的权重，b' 表示第二个全连接层的偏置。
其目标函数如下所示：

$$\begin{aligned}\arg\min_{\mathbf{W},\mathbf{W}',\mathbf{V},\mathbf{b},\mathbf{b}'}\ &\frac{1}{U}\sum_{u=1}^{U}\mathbb{E}_{p(\tilde{y}_u|y_u)}[\ell(\tilde{y}_u,\hat{y}_u)]\\ &+\mathcal{R}(\mathbf{W},\mathbf{W}',\mathbf{V},\mathbf{b},\mathbf{b}'),\end{aligned} \tag{5}$$

$$\mathcal{R}(\cdot)=\frac{\lambda}{2}(||\mathbf{W}||_2^2+||\mathbf{W}'||_2^2+||\mathbf{V}||_2^2+||\mathbf{b}||_2^2+||\mathbf{b}'||_2^2). \tag{6}$$

其中 $\mathbf{W}$ 和 $\mathbf{W}'$ 为两个全连接层的权重，$\mathbf{b}$ 和 $\mathbf{b}'$ 为两个全连接层的偏置，$\mathbf{V}$ 表示为每个用户分配的独立的神经元的权重，λ 为正则项在目标函数中的权重，$\mathbf{U}$ 表示用户的数目，$\ell(\cdot)$ 表示损失函数，使用的是均方误差损失函数。通过比较 CDAE[6]和 AutoRec[5]的目标函数不难发现两者的差距仅仅是 CDAE[6]多了一个神经元和偏置。为每一个用户分配一个不共享参数的神经元能够有效地防止模型发生过拟合从而导致模型坍塌。

Mult-VAE[7] 是第一个将变分自编码器用于协同过滤的模型，其模型如图 3所示。在介绍该模型之前需要知道什么是似然与概率。给定两个输入 θ 和 x 分别表示模型的参数和具体的数据。$P(x|\theta)$ 有以下两种情况：

- 如果 θ 是已知确定的，x 是变量，这个函数就叫作概率函数，它描述对于不同的样本点 x，其出现的概率。
- 如果 x 是已知确定的，θ 是变量，这个函数就叫做似然函数，它描述对于不同的模型参数，出现 x 这个样本点的概率是多少。

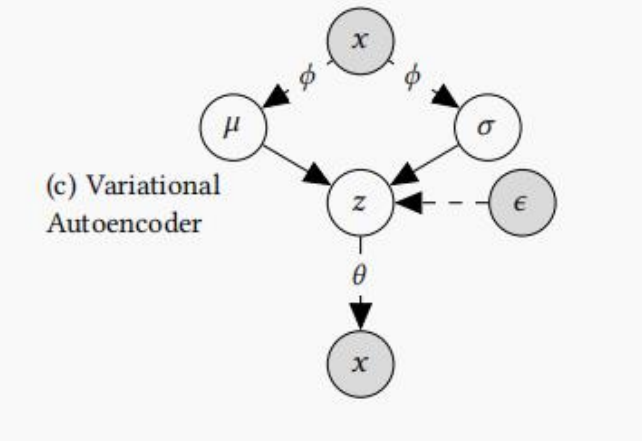

图 3 输入 x 为一个大小为 $|U|\times|I|$ 大小的评分矩阵，其经过一个全连接层之后得到两个相同的输出 μ 和 σ，然后对于 μ 不做改动，对于 σ 乘以一个服从某种分布的 ϵ，然后将其与 μ 相加得到 z，最后 z 再经过一个全连接层之后输出 x。

$$z_u \sim N(0, I_K), \pi(z_u) \propto \exp f_\theta(z_u),$$
$$x_u \sim \mathbf{Mult}(N_u, \pi(z_u)). \tag{7}$$

其中 $u \in \{1, ..., U\}$ 表示不同的用户，$i \in \{1, ..., I\}$ 表示不同的商品，$f_\theta(\cdot) \in \mathbb{R}^I$ 表示一个非线性的激活函数，且是在一个参数为 θ 的多层感知机之后的激活函数。$\pi(z_u)$ 表示经过归一化函数之后所得的概率向量。其包含所有的商品集合。
用户 u 的对数似然函数可以表示为：

$$\log p_\theta(x_u|z_u) \overset{c}{=} \sum_i x_{ui} \log \pi_i(z_u), \tag{8}$$

其目标函数为：

$$\mathcal{L}_\beta(x_u; \theta, \phi) = \mathbb{E}_{q\phi(z_u|x_u)}[\log p_\theta(x_u|z_u)]$$
$$-\beta \cdot \mathbf{KL}(q\phi(z_u|x_u)||p(z_u)). \tag{9}$$

其中，θ 表示所有的模型参数，$KL(a||b)$ 用于度量两个分布 a 与 b 之间的相似程度。β 是一个超参数，用于表示正则项在目标函数中所占权重。

算法 1 –VAE-SGD 使用随机梯度下降训练协同过滤变分自编码器

输入： 点击矩阵 $X \in \mathbb{R}^{U \cdot I}$ 随机初始化参数 θ, ϕ

WHILE 模型不收敛 **DO**
  采样一个批量大小的用户 U
  FOR $\forall u \in U$ **DO**
    采样 $\epsilon \sim N(0, I_K)$ 并通过重参数技巧计算 z_u
    通过 z_u 计算梯度 $\nabla_\theta \mathcal{L}$ 和 $\nabla_\phi \mathcal{L}$
  END FOR
  根据每个批次的梯度使用随机梯度下降更新 θ 和 ϕ
END WHILE
RETURN θ, ϕ

Bi-VAE[8] 对 Mult-VAE[7] 进行了改进，其模型所图 4所示。该模型的目标函数为：

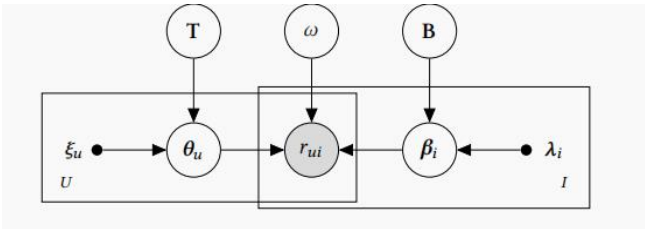

图 4 输入为一个大小为 $|U|\times|I|$ 大小的评分矩阵，实则是一个二维矩阵，先前的工作包括 VAE[7] 在内都只考虑了单一的维度的关系，要么只考虑了二维矩阵中的纵向关系，要么只考虑了横向的关系，BiVAE[8] 同时考虑了两种关系。如图所示，U 表示以用户为出发点，即考虑了矩阵中横向关系，I 表示以商品为出发点，即考虑了矩阵中的纵向关系。

$$\mathcal{L} = \sum_{u,i} \mathbb{E}_{q(\theta_u|r_{u*})} \mathbb{E}_{q(\beta_i|r_{*i})} [\log p(r_{ui}|\theta_u, \beta_i)]$$
$$- \sum_u \mathbf{KL}(q(\theta_u|r_(u*))||p(\theta_u)) - \sum_i \mathbf{KL}(q(\beta_i|r_{*i})||p(\beta_i)). \tag{10}$$

其中 θ_u 和 β_i 分别表示对用户和对商品分别编码的模型的参数。因为同时考虑了横向和纵向的关系，所以在求解正则的时候需要两个度量函数。
对于用户和商品两部分的优化可以进一步细分为以下两个公式：

$$\tilde{\mathcal{L}}^u = \sum_u [\log p(r_{u*}|\tilde{\theta}_u, \tilde{\beta}_{1:I}) - \mathbf{KL}(q(\theta|r_{u*})||p(\theta_u))]. \tag{11}$$

$$\tilde{\mathcal{L}}^i = \sum_u [\log p(r_{*i}|\tilde{\beta}_i, \tilde{\theta}_{1:U}) - \mathbf{KL}(q(\beta|r_{*i})||p(\beta_u))]. \tag{12}$$

算法 2 –Bi-VAE 的随机优化

输入： $\mathbf{R}, \xi, \lambda, g_\omega, \tilde{\mu}_{\tilde{\psi}}, \tilde{\sigma}_{\tilde{\psi}}, \tilde{\mu}_{\tilde{\phi}}, \tilde{\sigma}_{\tilde{\phi}}$， 批量大小 m

输出： $\omega, \tilde{\psi}, \tilde{\phi}$

REPEAT
  基于用户目标的优化
  从观测变量中采样一部分 $\{r_{*1}, ..., r_{*I}\}$
  从后验 $q(\beta|r)$ 中采样一部分 $\{\tilde{\beta}_1, ..., \tilde{\beta}_I\}$
  从观测变量中采样一部分 $\{r_{1*}, ..., r_{m*}\}$
  从后验 $q(\theta|r)$ 中采样一部分 $\{\tilde{\theta}_1, ..., \tilde{\theta}_m\}$
  使用梯度下降更新参数 ω, ψ
  基于商品目标的优化
  从观测变量中采样一部分 $\{r_{1*}, ..., r_{U*}\}$
  从后验 $q(\theta|r)$ 中采样一部分 $\{\tilde{\theta}_1, ..., \tilde{\theta}_U\}$
  从观测变量中采样一部分 $\{r_{*1}, ..., r_{*m}\}$
  从后验 $q(\beta|r)$ 中采样一部分 $\{\tilde{\beta}_1, ..., \tilde{\beta}_m\}$
  使用梯度下降更新参数 $\omega, \tilde{\psi}$
UNTIL 模型收敛

3 实验与结论

3.1 实验

3.1.1 数据集

数据集使用的是公开的数据集 ML-1M[1]，该数据集经过数据预处理之后的数据统计如表 1所示。其中用户数目为 6040，商品数目为 3706，如果每个用户都交互了所有的商品的情况下，评分矩阵大小应该为 22384240，但是实际交互数据只用 1000209，只占比 4.46%，数据还是相当稀疏的。

数据名称	用户数目	商品数目	评分数目	密集性
ML-1M	6040	3706	1000209	4.47%

表 1 ML-1M 数据统计表

3.1.2 评价指标

评价指标采用的是召回率和归一化折损累计增益，两者公式如下：

$$\mathbf{Recall}_u = \frac{|\mathbf{R}(u) \cap \mathbf{T}(u)|}{|\mathbf{T}(u)|}. \tag{13}$$

其中 $\mathbf{R}(u)$ 表示预测值，$\mathbf{T}(u)$ 表示真实值，召回率就是关注最后预测正确商品的概率。

$$\mathbf{NDCG} = \frac{\mathbf{DCG}}{\mathbf{IDCG}}, \tag{14}$$

$$\mathbf{DCG} = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log_2(i+1)}, \tag{15}$$

$$\mathbf{IDCG} = \sum_{i=1}^{|\mathbf{REL}|} \frac{2^{rel_i} - 1}{\log_2(i+1)}. \tag{16}$$

因为在实验中只是做了预测下一个商品，所以 **IDCG** 是始终为 1 的,所以实际 **NDCG** 就等于 **DCG**。归一化折损累计增益考虑了位置的关系，就是预测值中命中真实值的位置。

3.1.3 实验结果

本实验选取的模型为基于邻域的协同过滤模型[3],包括 ItemCF(基于商品的协同过滤)和 UserCF(基于用户的协同过滤)。IitemCF 是指使用商品来进行协同过滤，简单来说就是选取与用户最后点击的商品最相似的商品进行推荐，这个相似性度量函数如下所示：

$$W_{ij} = \frac{|N(i) \cap N(j)|}{\sqrt{|N(i)||N(j)|}}. \tag{17}$$

其中 W_{ij} 表示物品 i 与物品 j 的相似程度，$|N(i)|$ 表示物品 i 出现的次数，$|N(j)|$ 表示物品 j 出现的次数，$|N(i) \cap N(j)|$ 表示物品 i 与物品 j 共同出现的次数。

UserCF 是指使用用户来进行协同过滤，简单来说就是选取与目标用户最相似的其他用户中目标用户没有交互过的商品进行推荐，这个相似性度量函数如下所示：

$$p(u,i) = \sum_{v \in S(u,K) \cap N(i)} W_{uv} r_{vi}. \tag{18}$$

其中 $S(u,K)$ 表示与目标用户 u 最相似的 K 个用户构成的集合，$N(i)$ 表示目标用户没有交互过的商品，$W_{uv} r_{vi}$ 表示用户对于商品 i 的评分。

还有基于变分自编码器的两个协同过滤模型 Mult-VAE[7] 和 BiVAE[8]。实验结果如表所示。

数据集	评价指标	ItemCF	UserCF	Multi-VAE	BiVAE
ML-1M	Recall@5	0.0444	0.0267	0.0372	0.0264
	Recall@10	0.0775	0.0467	0.0706	0.0529
	Recall@20	0.121	0.0849	0.1221	0.0966
	Recall@50	0.2228	0.173	0.2389	0.1952
	NDCG@5	0.0287	0.0172	0.1295	0.1038
	NDCG@10	0.0392	0.0236	0.1331	0.1075
	NDCG@20	0.0501	0.0332	0.144	0.1169
	NDCG@50	0.0699	0.0507	0.1836	0.1492

表 2 模型实验结果

3.1.4 结果分析

通过比较 ItemCF 与 UserCF 两列可知，基于商品的协同过滤是好于基于用户的协同过滤的，因为 UserCF 在考虑用户的相似性的时候仅仅考虑了用

[1] https://grouplens.org/datasets/movielens

户之间商品的交集的数目所占的比例，并没有考虑这些物品的位置关系。例如两个用户分别交互了 10 个商品，但两个用户有交集的商品只是前两个，说明用户对于后面 8 个商品的意图是不一样的，直接通过两者的并集进行推荐并不会有一个好的效果。相反 ItemCF 考虑给用户推荐的是与其点击的最后一个商品最相似的未交互过的商品，这就考虑了用户最近的意图。

通过比较 ItemCF 与 Mult-VAE 的实验结果可知，基于商品的协同过滤在 Recall@5 和 Recall@10 这两个指标上是好于 Mult-VAE 的，可能是因为其考虑了用户最近的意图，但是其在 NDCG 的所有指标上都是比 Mult-VAE 差的，可能是因为其预测的商品的位置偏于后面。

通过比较实验结果可知，基于变分自编码器的协同过滤模型[7,8] 在折损累计增益指标上是完全好于基于邻域的传统协同过滤模型[3] 的。可能是因为传统的模型只考虑邻域的关系，而没有考虑全局的关系。

4 展望与结论

4.1 展望

协同过滤推荐系统主要依赖于用户的历史交互数据，并且会遭遇冷启动问题。为了缓解这个问题，一些研究工[14–16] 提出使用辅助信息，例如用户画像和商品属性来丰富用户与商品的表示。除此之外，还有一些使用更加高效的学习机制来缓解对数据的重度依赖，例如小样本学习[16,17]。与此同时，在协同过滤推荐模型中，一直以来都面临者数据稀疏的问题，因为用户与商品的交互数据是远远小于两者的数目的乘积的。传统的协同过滤模型要么只考虑单一的线性关系，要么没有充分利用原始数据中的潜在关系。虽然实现的方法简单易懂，但是受限于数据稀疏。随着神经网络的快速发展，利用神经网络对商品和用户进行编码，这在一定程度上缓解了数据稀疏的问题。通过神经网络对商品和用户进行编码，并从细粒度考虑数据的编码信息可能会成为未来协同过滤研究工作中的重点。

4.2 结论

通过论文中对于模型的分析和实验结果的比较，可以知道通过神经网络可以缓解协同过滤中数据稀疏的问题，通过细粒度地利用原始的数据，可以进一步缓解数据稀疏的问题。例如在编码过程中考虑用户的独特性[6]，在对商品进行编码时考虑数据的分布[7]，分别对用户和商品进行编码[8]。这些工作都很好地缓解了数据稀疏的问题。

参考文献

[1] KOREN Y, BELL R, VOLINSKY C. Matrix factorization techniques for recommender systems[J]. Computer, 2009, 42(8):30-37.

[2] LEE J, KIM S, LEBANON G, et al. Local low-rank matrix approximation[C]//International conference on machine learning. [S.l.]: PMLR, 2013: 82-90.

[3] SARWAR B, KARYPIS G, KONSTAN J, et al. Item-based collaborative filtering recommendation algorithms[C]//Proceedings of the 10th international conference on World Wide Web. [S.l.: s.n.], 2001: 285-295.

[4] SALAKHUTDINOV R, MNIH A, HINTON G. Restricted boltzmann machines for collaborative filtering[C]//Proceedings of the 24th international conference on Machine learning. [S.l.: s.n.], 2007: 791-798.

[5] SEDHAIN S, MENON A K, SANNER S, et al. Autorec: Autoencoders meet collaborative filtering[C]//Proceedings of the 24th international conference on World Wide Web. [S.l.: s.n.], 2015: 111-112.

[6] WU Y, DUBOIS C, ZHENG A X, et al. Collaborative denoising auto-encoders for top-n recommender systems[C]//Proceedings of the ninth ACM international conference on web search and data mining. [S.l.: s.n.], 2016: 153-162.

[7] LIANG D, KRISHNAN R G, HOFFMAN M D, et al. Variational autoencoders for collaborative filtering[C]//Proceedings of the 2018 world wide web conference. [S.l.: s.n.], 2018: 689-698.

[8] TRUONG Q T, SALAH A, LAUW H W. Bilateral variational autoencoder for collaborative filtering[C]//Proceedings of the 14th ACM International Conference on Web Search and Data Mining. [S.l.: s.n.], 2021: 292-300.

[9] BENGIO Y, COURVILLE A, VINCENT P. Representation learning: A review and new perspectives[J]. IEEE transactions on pattern analysis and machine intelligence, 2013, 35(8):1798-1828.

[10] GOPALAN P, HOFMAN J M, BLEI D M. Scalable recommendation with hierarchical poisson factorization.[C]//UAI. [S.l.: s.n.], 2015: 326-335.

[11] HU Y, KOREN Y, VOLINSKY C. Collaborative filtering for implicit feedback datasets[C]//2008 Eighth IEEE international conference on data mining. [S.l.]: Ieee, 2008: 263-272.

[12] RENDLE S, FREUDENTHALER C, GANTNER Z, et al. Bpr: Bayesian personalized ranking from implicit feedback[J]. arXiv preprint arXiv:1205.2618, 2012.

[13] MNIH A, SALAKHUTDINOV R R. Probabilistic matrix factorization [J]. Advances in neural information processing systems, 2007, 20.

[14] MA H, ZHOU D, LIU C, et al. Recommender systems with social regularization[C]//Proceedings of the fourth ACM international conference on Web search and data mining. [S.l.: s.n.], 2011: 287-296.

[15] MANOTUMRUKSA J, MACDONALD C, OUNIS I. Regularising factorised models for venue recommendation using friends and their comments[C]//Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. [S.l.: s.n.], 2016: 1981-1984.

[16] YU J, GAO M, LI J, et al. Adaptive implicit friends identification over heterogeneous network for social recommendation[C]//Proceedings of the 27th ACM international conference on information and knowledge management. [S.l.: s.n.], 2018: 357-366.

[17] LI J, JING M, LU K, et al. From zero-shot learning to cold-start recommendation[C]//Proceedings of the AAAI conference on artificial intelligence: volume 33. [S.l.: s.n.], 2019: 4189-4196.

[18] VINCENT P, LAROCHELLE H, BENGIO Y, et al. Extracting and composing robust features with denoising autoencoders[C]//Proceedings of the 25th international conference on Machine learning. [S.l.: s.n.], 2008: 1096-1103.

附录 A 附录

A.1 协同过滤案例

图 5 图中包含三个用户不同的交互历史，分别对应了 User1,User2,User3。由图像可以看出，User1 与 User2 交互历史的交集包括手表和领带，User1 和 User3 交互历史的交集包括领带和皮鞋，User2 和 User3 交互历史的交集包括领带。

由上图可知，根据商品的协同过滤原理，商品手表和商品皮带是相似的，根据公式 17，可以获知两者相似性为 $\frac{2}{\sqrt{6}}$，同理可得领带和皮鞋的相似性为 $\frac{2}{\sqrt{6}}$，最后将所有的物品相似性计算之后会给每个用户推荐与其最后一个商品最相似的但其没有交互过的商品。例如会给 User2 推荐皮鞋（因为与领带最相似的物品中皮鞋是 User2 没有交互过的）。根据公式 18，可以获知 User1 与 User2 的相似度为 $\frac{2}{12-2}=\frac{1}{5}$，User1 与 User3 的相似度为 $\frac{2}{9}$，User2 与 User3 的相似度为 $\frac{1}{10}$，然后根据用户协同过滤的理论会为 User1 推荐 User3 中其没有交互过的商品例如裙子、画盘、书籍等。

A.2 KL 散度

KL(Kullback-Leibler divergence) 散度又称相对熵 [2]，是用来描述两个概率分布 **P** 和 **Q** 的差异的一种办法。其具有非对称性，即 $D(\mathbf{P}||\mathbf{Q}) \neq D(\mathbf{Q}||\mathbf{P})$。在信息论中，$D(\mathbf{P}||\mathbf{Q})$ 表示用概率分布 **Q** 来拟合真实分布 **P**，产生的信息损耗。**KL** 散度公式定义如下：

对于离散型变量有：

$$D(\mathbf{P}||\mathbf{Q}) = \sum_{i \in X} P(i) * [\log(\frac{P(i)}{Q(i)})] \tag{19}$$

对于连续性变量有：

$$D(\mathbf{P}||\mathbf{Q}) = \int_{x} P(x) * [\log(\frac{P(i)}{Q(i)})] dx \tag{20}$$

相对熵是大于 0 的，因为 $-\log x$ 是凸函数，根据吉布斯不等式 [3] 和相对熵的定义有：

$$\begin{aligned} D(\mathbf{P}||\mathbf{Q}) = \sum_{i \in X} P(i) * [\log(\frac{P(i)}{Q(i)})] = -E[log(\frac{Q(i)}{P(i)})] \geqslant \\ -log[\sum_{i \in X} P(i) * \frac{Q(i)}{P(i)}] = -\log[\sum_{i \in X} Q(i)] = 0 \end{aligned} \tag{21}$$

由上式可知，相对熵是恒大于等于 0 的。当且仅当两个分布相同时，相对熵等于 0。

[2] https://baike.baidu.com/item/%E7%9B%B8%E5%AF%B9%E7%86%B5/4233536

[3] https://baike.baidu.com/item/%E5%90%89%E5%B8%83%E6%96%AF%E4%B8%8D%E7%AD%89%E5%BC%8F/22780937